CD-doc-2345

SiD Simulations and Benchmarking

Rob Kutschke, CD/IDS

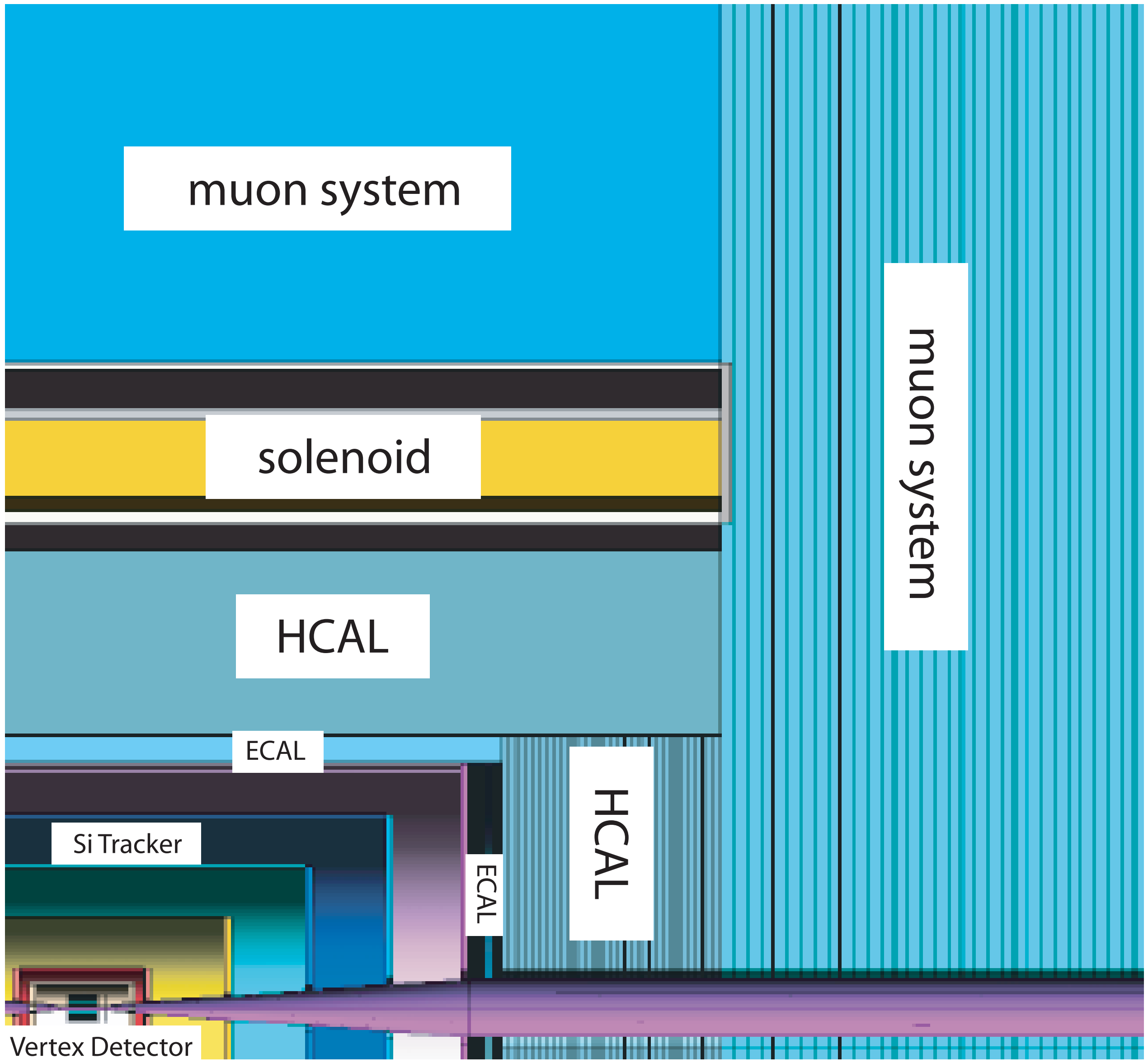
ILC Coordination Forum

July 17, 2007

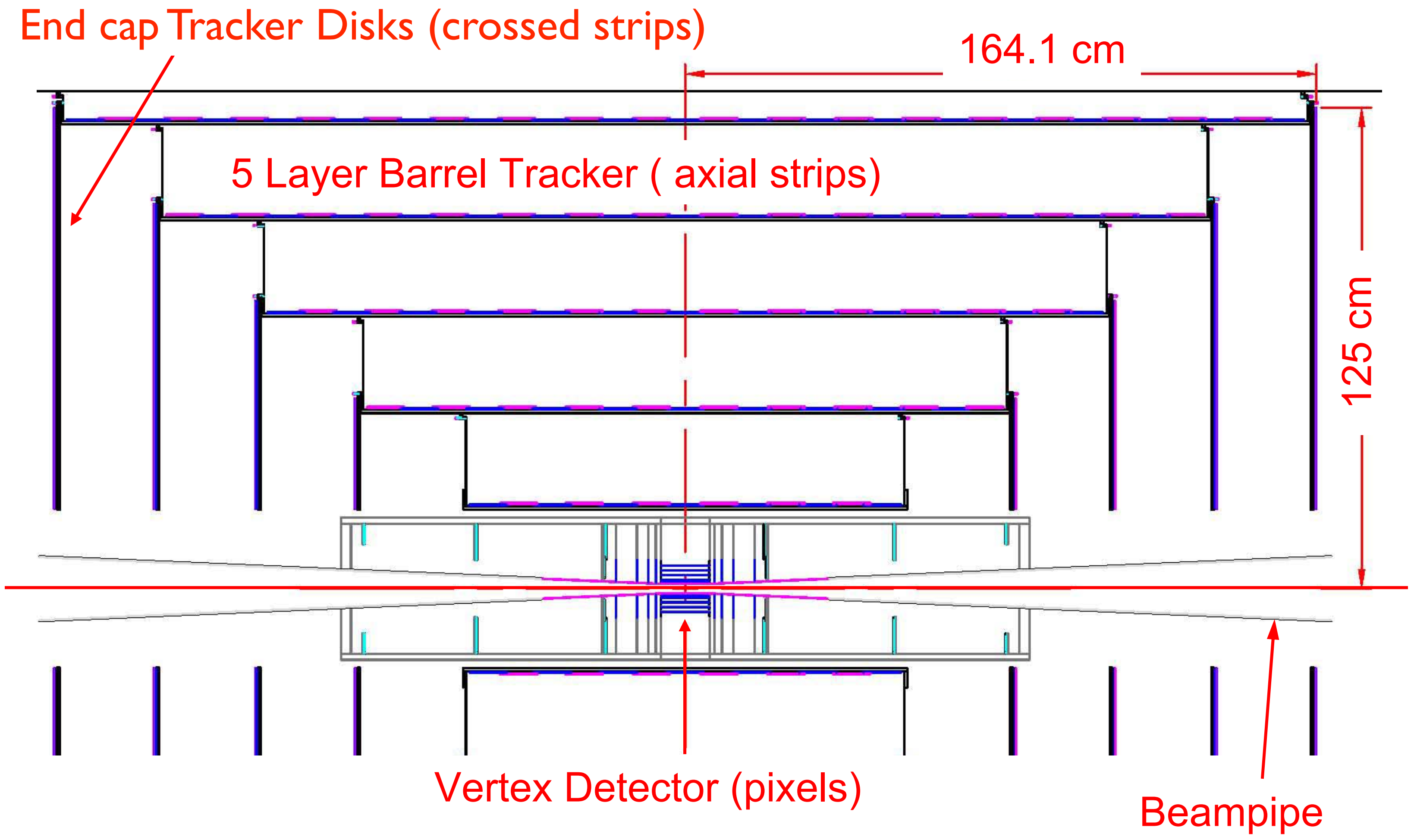
Outline

- SiD Detector
- SiD Software
- Work plan
  - Hans' talk will give more details on the forward tracking jobs.
- Miscellaneous
  - What Adam is up to (I hope I get this right ...)

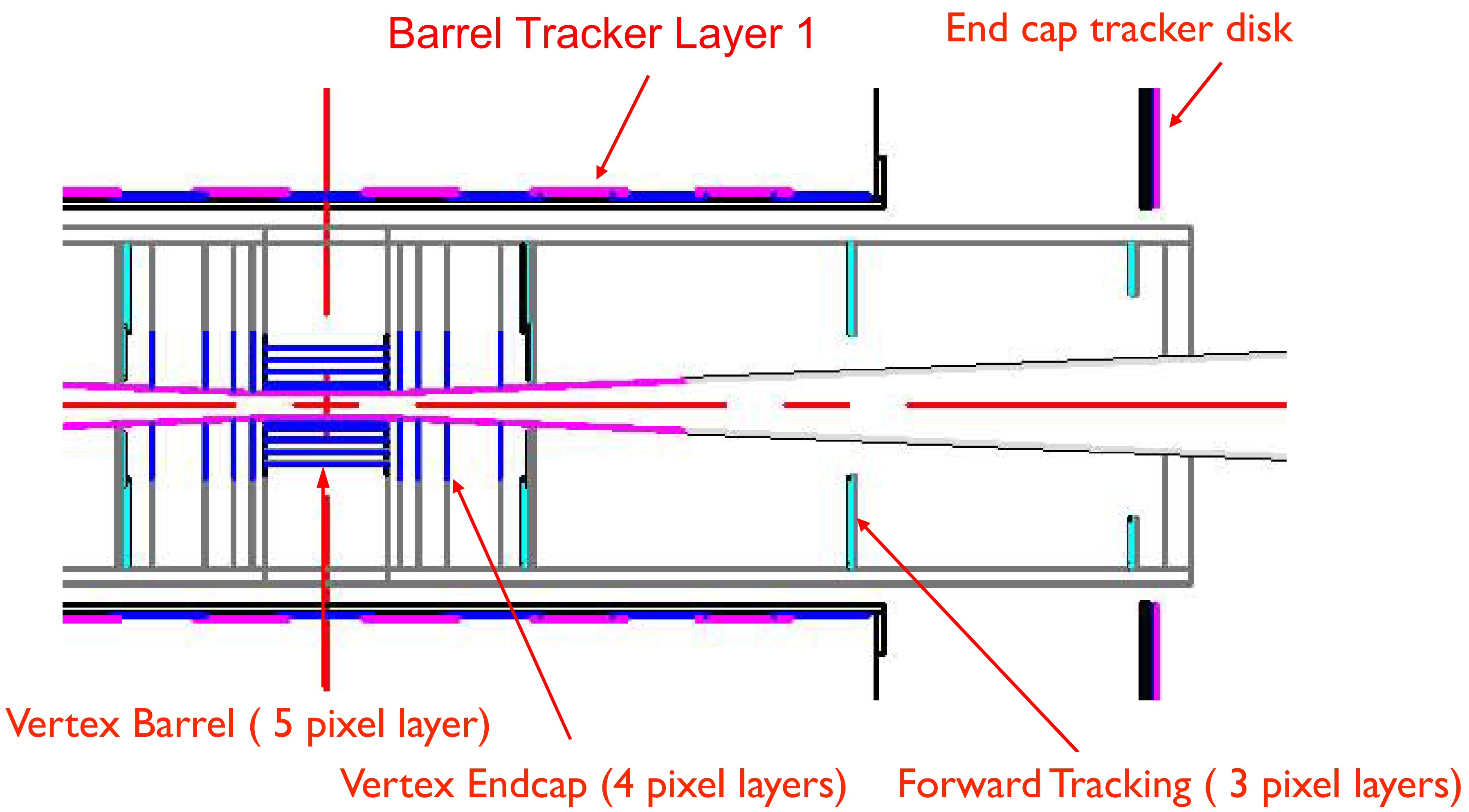
The SiD detector



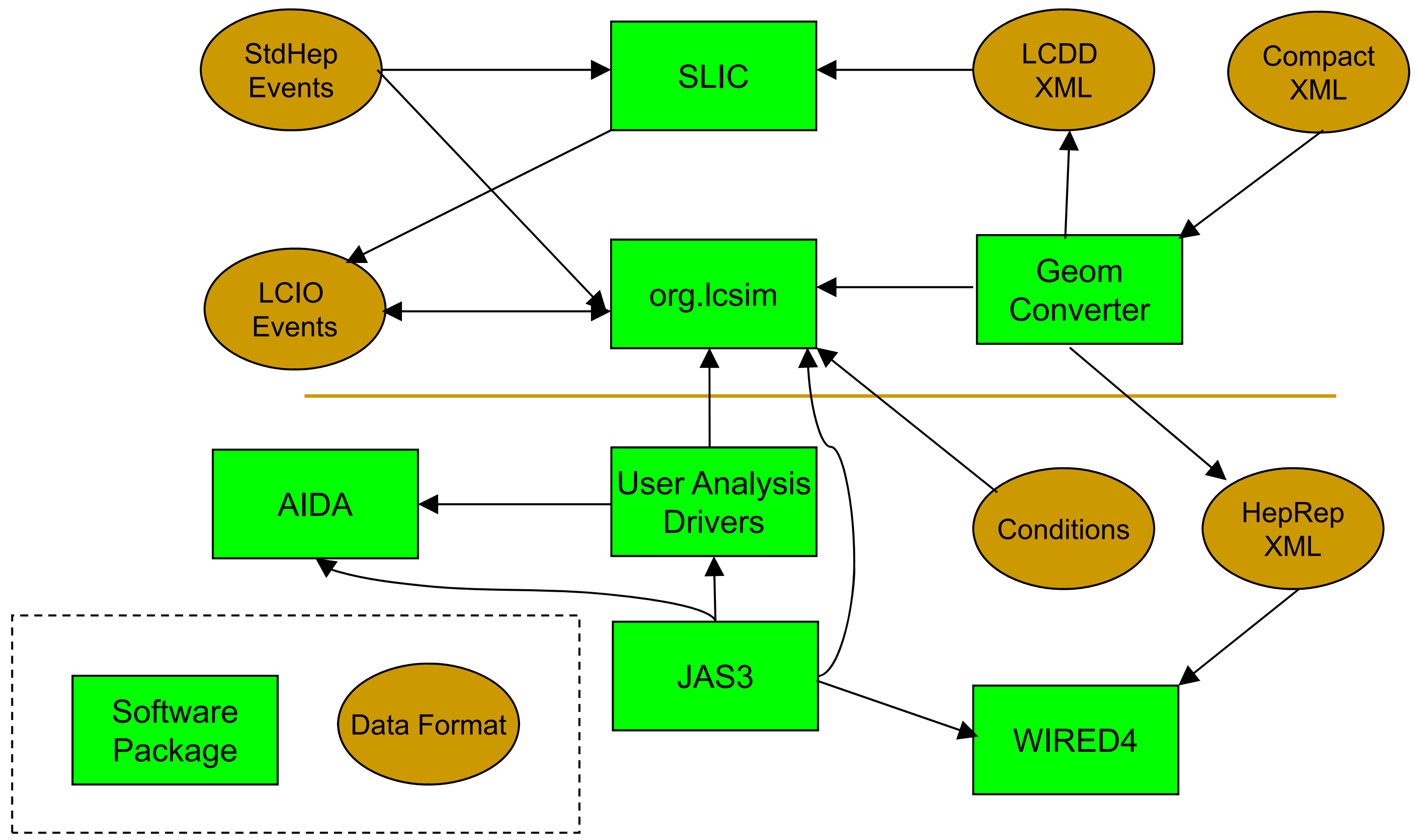
SiD Tracking System, Vertical Section


End cap Tracker Disks (crossed strips)
164.1 cm
5 Layer Barrel Tracker (axial strips)
125 cm
Vertex Detector (pixels)
Beampipe

Detail Near Beamline


Barrel Tracker Layer 1
End cap tracker disk
Vertex Barrel (5 pixel layer)
Vertex Endcap (4 pixel layers)
Forward Tracking (3 pixel layers)

SiD Software Overview



SiD Software

- SLIC:
  - C++. G4 based simulation system.
- org.lcsim
  - Java based framework for reconstruction and analysis.
- Wired 4 based event display.
- GeomConverter:
  - Reads “Compact Detector Description” XML.
  - Native format for org.lcsim.
  - Can write:
    - HepRep XML for Wired-4
    - LCDD XML for SLIC
- Data formats: StdHep and LCIO.

org.lcsim

- Java based.
- Not a full featured framework.
    - Good enough for a small group with documentation by lunch, coffee and beer:30.
- All key reconstruction codes live in user areas.
    - A loose collection of disconnected tools.
    - Historically user codes do not play well together.
        - Hit and Track classes are deficient so everyone makes private extentions.
- Native histogram/tuple environment:
    - aida + JAS3 as a viewer; much poorer than root.
- Weak release/distribution model.
- More details in backup pages.

SiD Detector Models

- sid00
  - Complete but simplified sensitive volumes.
    - Barrel vertexer and tracker are pure cylinders.
    - Endcaps are annulus of disk.
  - Lots of exisiting MC needed by PFA people.
- sid01
  - As above but more detail of dead material.
  - Added forward tracker.
  - Current official model.
- New model under development.
  - Tracking elements made from wafers.
  - Will need several variations, especially in forward region, including variations of dead material.

Our Jobs

- Forward Vertexing and Tracking:
  - Study occupancies using existing detector models.
  - Help to define the new detector model
    - Includes dead materials
  - Real track reconstruction in forward region.
    - Includes pattern recognition and fitting in presence of backgrounds.
- Simulated analyses.
- SiD @ FNAL web site.

Plus whatever infrastructure work is implied

Our Jobs - Next Level of Detail

1. Study occupancies, using existing models.
2. Bookkeeping and Infrastructure improvements:
   - Help to define new detector model.
   - See next page.
3. Get Kalman filter code working.
4. Enhancements to org.lcsim
   - Real pattern recognition in forward region.
   - "Port"/exercise existing code:
     - Vertexing/Jet Finding/Jet Flavor Id/
5. Simulated Analyses
   - B(H$\rightarrow$b bbar) and B(H$\rightarrow$c cbar).
6. FNAL web site.
   - Make it easy for new people to ramp up.

Bookkeeping and Infrastructure

- Classes that need to be fixed:
  - RawTrackerHit (sort of a digi)
  - Track
- New classes needed:
  - Clusters of digis and clustering algorithms.
  - Bookkeeping of used hits.
  - Collection of muon and electron candidates.
- We are waiting on code to create RawTrackerHits from SimTrackerHits (create digis from hits).
- We can create classes but not persist them!
  - Agitate for a new persistency model.
- Effort slowed by demand that all persistent classes be usable by all detector concepts.

Our Jobs - Who is doing What

1. Study occupancies, using existing models.
   - Fransisco supervised by Hans.
2. Bookkeeping and Infrastructure improvements:
   - Help to define new detector model.
     - Geometry back end being done at SLAC.
     - Hans and students: work with Bill Cooper for models of support and variations on the wafer layout.
   - Hit, Track and e/mu classes:
     - Rob and Hans with input from SLAC and others.
3. Get Kalman filter code working.
   - Rob

Our Jobs - Who is doing What

4. Enhancements to org.lcsim
   - Real pattern recognition in forward region.
     - Hans.
   - "Port"/exercise existing code:
     - Vertexing/Jet Finding/Jet Flavor Id/
     - Rob
5. Simulated Analyses
   - B(H$\rightarrow$b bbar) and B(H$\rightarrow$c cbar).
   - Rob
6. FNAL web site.
   - Lynn and Rob

Relevant Deadlines

- ALCPG October 22-26, 2007 at FNAL
  - First pass at one benchmark study for CDR.
- Spring 2008
  - Software for CDR benchmarks essentially complete.
  - CDR benchmark studies underway.
  - Start writing CDR.
- Fall 2008
  - Submit CDR.

Deadlines with Added Detail

- ALCPG October 22-26, 2007 at FNAL
  - Occupancy studies and most infrastructure done.
  - Kalman filter and other "ported" codes working
  - First release of detector built of wafers sometime in the summer.
  - First pass on one simulated analysis.
- Spring 2008
  - Our software working well enough for general use.
  - Several simulated analyses underway.
  - Start writing CDR.
- Fall 2008
  - Submit CDR.

Summary

- We have agreed to a list of jobs
  - Lots of forward tracking.
  - One simulated analysis.
  - Precursor infrastructure work is required too.
  - FNAL web site.
- We have a rough outline of who is doing what with specific deadlines for the October ALCPG meeting and less specific details for afterwards.

Backup Slides

org.lcsim

- Java based.
- Can be run standalone or within JAS3.
  - Documentation/examples are JAS3-centric.
- Framework runs the event loop and executes a list of "drivers" specfied by the user.
- Driver:
  - What other frameworks call a module.
  - Callable from the framework:
    - Detector change; process event; end of data ...
  - Can read event and add collections to the event.
  - Can overwrite/delete collections in an event.
- Native histogram/tuple environment: aida.
  - Display tools not as rich as root.

org.lcsim (2)

- Reconstruction code lives in user areas and is not vetted by anyone.
- Little discipline among users to ensure that their codes cooperate.
  - Predefined classes are not rich enough for the job.
    - So everyone makes their own private extensions.
    - Can add these objects to the event - but no persistency.
- No method to stop my histograms or collections from stomping on yours.
- Various "full" reconstruction codes are advertised:
  - Some ran in JAS2 and are not yet ported to JAS3.
  - Documentation by calling the author.
  - I have not yet run any of them.

org.lcsim (3)

- Release model
  - Infrequent releases.
  - Users: copy current .jar files from SLAC
  - Developers: build the head
    - You just gotta know when the head is/was in good shape.
- Each user keeps current .jar files in ~/.JAS3
  - Deploying a new release clobbers the old one and you cannot backtrack unless you saved a copy ahead of time or know the check out time of the old version.
- Presumes that you always have internet access.
  - It does cache things but you may need to know in advance if you need to force caching.